Jiménez-Moreno, Amaya, Martínez-Enríquez, Eduarado, Díaz-de-María, Fernando. (2016). Complexity Control based on a Fast Coding Unit Decision Method in the HEVC Video Coding Standard. *IEEE Transactions on Multimedia*, 18(4), pp.: 563 - 575.

DOI: https://doi.org/10.1109/TMM.2016.2524995

Complexity Control based on Fast Coding Unit Decision in the HEVC Video Coding Standard

Amaya Jiménez-Moreno, Eduardo Martínez-Enríquez, and Fernando Díaz-de-María, *Member, IEEE*

***Abstract*—The emerging High Efficiency Video Coding standard (HEVC) achieves higher coding efficiency than previous standards thanks to a set of new coding tools such as the quadtree coding structure. With this novel structure, the pixels are organized in Coding Units (CU), then in Prediction Units (PU), and finally in Transform Units (TU), whose sizes can be optimized at every level following a tree configuration. These tools allow a very flexible data representation; however, they involve a very high computational complexity.**

In this paper we propose an effective complexity control (CC) algorithm. Our proposal is based on a hierarchical approach. An early termination condition is set at every CU size to decide if the following CU sizes should be explored or not. The actual encoding times are also considered in order to meet the target complexity in real-time. Moreover, all the parameters of the algorithm are estimated *on-the-fly* to adapt its behavior to the content, the encoding configuration, and the target complexity over time.

The proposed method has been experimentally tested for a large variety of video sequences and coding configurations. The experimental results prove that our proposal is able to meet target complexities reductions of up to 60% with respect to a full-exploration, with a notable accuracy and quite limited losses in coding performance. Moreover, it has been compared with a complexity control state-of-the-art method, achieving a significantly better trade-off between coding complexity and efficiency, besides higher accuracy to meet the target complexity. Also, a comparison with a complexity reduction state-of-the-art method highlights the advantages of the complexity control framework. Finally, we show that the proposed method works fine when the target complexity varies over time, which is a desirable feature for devices with changing computational resources.

***Index Terms*—Complexity control, fast coding unit decision, HEVC, *on-the-fly* estimation.**

I. Introduction

The High Efficiency Video Coding (HEVC) is the latest video coding standard developed by the Joint Collaborative Team on Video Coding (JCT-VC). The current popularity of high definition (HD) video signals, or even higher resolutions, and the huge amount of new applications based on video services, such as video streaming in Internet and mobile networks, are driving the need of video coding standards with higher coding efficiency than the previous ones. The HEVC standard deals with these new needs. HEVC is a block-based hybrid coding technique that significantly outperforms previous video coding standards, such as H.264/AVC, MPEG-4 or MPEG-2. It is able to duplicate the coding efficiency when compared to H.264/AVC [1] thanks to a set of new coding techniques included in HEVC [2]. The most relevant new feature of the HEVC standard is the quadtree coding structure. In previous standards, every frame in the video sequence was divided in macroblocks (MBs), which are 16×16 pixels blocks. In HEVC, each frame is divided in blocks of equal size called coding tree blocks (CTBs), whose size can be selected as 16×16, 32×32, or 64×64 pixels. Then, HEVC allows dividing the CTBs into smaller coding units (CUs) using a quadtree coding structure. The dimensions of the CUs can be from 8×8 pixels up to the CTB dimensions, depending on the tree depth at which the CU is located. The higher depth, the lower the CU dimensions. Therefore, a CTB can consist of either only one CU or multiple CUs.

Each CU can in turn be divided into prediction units (PUs) and transform units (TUs). Regarding the prediction stage, every CU can be further split into PUs of various sizes, on which the motion estimation (ME) process is performed. Specifically, each CU either could not be further divided or could be divided into two or four blocks, which could be of asymmetric sizes. Fig. 1 shows all the partitioning possibilities of a CU of 2N×2N pixels into PUs.

Regarding the transform stage, the CU can be considered as the root of another quadtree in charge of carrying out the transform process of the residue obtained after the prediction stage. The sizes of the TUs can vary from 4×4 up to the CU size, depending on the depth at which the TU is located. The left side of the Fig. 2 shows an example where a CTB of 64×64 pixels is divided into the corresponding CUs (in solid line) and TUs (in dashed line). In the right part of Fig. 2, the associated quadtree is shown.

The encoder needs to select the best coding option among all the possibilities previously explained. The best coding quadtree structure (including the CUs, PUs, and TUs sizes) is selected through a Rate-Distortion Optimization (RDO) process, which must evaluate every tree configuration and compare all of them in terms of rate and distortion. Specifically, the encoder seeks the coding option that minimizes a distortion measure subject to a given rate constraint:

$$\min_{\boldsymbol{\theta}} \{D(\boldsymbol{\theta})\} \text{ subject to } R(\boldsymbol{\theta}) \leq R_c, \tag{1}$$

where $\boldsymbol{\theta}$ represents a combination of different coding options (CU, PU, TU, motion vectors (MVs), etc.); $D(\boldsymbol{\theta})$ represents

EDICS: Signal Processing for Multimedia Applications ⟶ Compression and Coding (CPRS).

This work has been partially supported by the National Grant TEC2014-53390-P of the Spanish Ministry of Economy and Competitiveness.
Amaya Jiménez-Moreno, and Fernando Díaz-de-María are with the Department of Signal Theory and Communications, Carlos III University, Leganés (Madrid), Spain (e-mail: ajimenez@tsc.uc3m.es, fdiaz@tsc.uc3m.es). Eduardo Martínez-Enríquez is with the Instituto de Óptica, Consejo Superior de Investigaciones Científicas, Madrid, Spain (e-mail: eduardo.martinez@io.cfmac.csic.es).

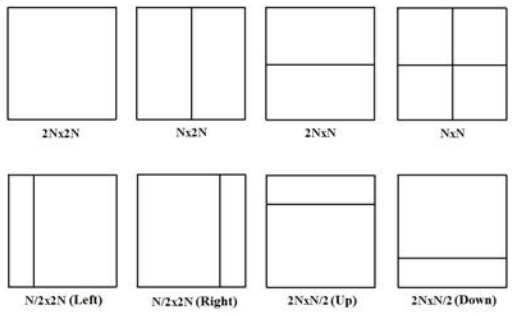


Fig. 1. PU sizes for a CU of 2N×2N pixels.

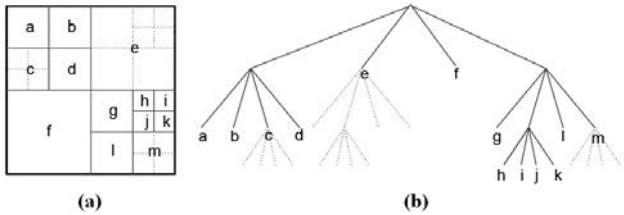


Fig. 2. (a) In solid line an example of a 64×64 CTB divided into 13 CUs. Two CUs (e, and f) of 32×32, seven CUs (a, b, c, g, l, and m) of 16×16, and four CUs (h, i, j, and k) of 8×8 pixels. In dashed line an example of a TU structure where the TU size can be the same as of the CU (a, b, d, f, g, l, h, i, j, and k) or not (c, e, and m). (b) The associated quadtree, in solid line for CUs and in dashed line for TUs.

the distortion between the original and the reconstructed CU; and $R(\boldsymbol{\theta})$ is the rate needed to encode the CU, both using $\boldsymbol{\theta}$; finally, R_c represents the rate constraint.

This problem can be solved using Lagrange formulation, which leads to the following unconstrained problem [3]:

$$\min_{\boldsymbol{\theta}} \{J\} \text{ with } J(\boldsymbol{\theta}) = D(\boldsymbol{\theta}) + \lambda R(\boldsymbol{\theta}), \tag{2}$$

where $J(\boldsymbol{\theta})$ represents the cost associated with a set of coding options $\boldsymbol{\theta}$; and λ is the Lagrange multiplier that weights the relative importance between $D(\boldsymbol{\theta})$ and $R(\boldsymbol{\theta})$. Thus, a certain λ value yields a solution $\boldsymbol{\theta}^*(\lambda)$ that turns out to be an optimal solution to the original RDO problem (1) for $R_c = R(\boldsymbol{\theta}^*)$.

$R(\boldsymbol{\theta})$ is calculated as the total bit rate, including the information related with the residual transform coefficients, the MVs, etc., considering the CU, PU, and TU sizes decisions. $D(\boldsymbol{\theta})$ is calculated as a sum of squared errors (SSE) between the original and reconstructed blocks. Therefore, to calculate $R(\boldsymbol{\theta})$ and $D(\boldsymbol{\theta})$ the encoder requires carrying out the prediction, residue calculation, transformation, quantization, entropy coding, and the inverse processes with every possible configuration. This process results in extremely high computational complexity, which actually is far above the complexity of the H.264/AVC standard and becomes the bottleneck of the HEVC standard. Thus, the motivation of this work is to design an algorithm able to control the computational complexity of a HEVC video encoder and, consequently, allow for more efficient implementations of the HEVC standard.

Our proposal is focused on the complexity control problem, as there are just few works related with this subject and further improvements still can be achieved. Using the statistical properties of the sequences and the time measures of the encoder we will be able to adjust the encoding process to the required target computational burden. Moreover, we propose to adaptively adjust the parameters of our method, trying to avoid the generalization problem associated to the use of fixed values in the performance parameters.

The rest of this paper is organized as follows. In Section II an overview of the state-of-the-art methods that deal with the complexity control (CC) and complexity reduction for the HEVC standard is presented, as well as the main contributions of our proposal. Section III provides a detailed explanation of the proposed method. In Section IV the experimental results supporting the proposal are presented and discussed. And finally, Section V summarizes our conclusions and outlines future lines of research.

II. Review of the State-of-the-Art

Several works have been published that address the problem of the high computational complexity of the HEVC standard. Two related tasks can be identified in this field: CC and complexity reduction. In the first case, the encoder must meet a specific target complexity, assuring that the encoding process fits the available computational resources. On the other hand, in the second case the goal is just to reduce the complexity of the encoding process as much as possible. As explained before, the optimal representation of a CTB can be broken down into a series of decisions related to CU depths, PU modes, and TU quadtree sizes. Thus, both the complexity reduction or the complexity control problems have been usually addressed by providing fast solutions to different sub-problems dealing with determining CU depths, PU modes, or TU sizes, or a combination of them.

Since our goal is to design a method to control the complexity of an HEVC encoder based on a fast CU depth determination, hereafter we focus the description of the state-of-the-art on the methods for fast CU depth determination and on the CC problem itself. Therefore, although there exist relevant works dealing with ways to achieve efficient solutions to the PU mode selection (e.g., [4]–[7]) or to the TU size selection (e.g., [8] and [9]), they are not covered here.

In [10] a CC method for HEVC was proposed. The method relies on the observation that in those co-located regions of consecutive frames which kept some features unaltered (motion, texture, etc.), the CU depths selected as optimal tended to maintain. Thus, this information can be used in the next frames, which the authors called "constrained frames", avoiding the assessing of the remaining CUs. The CC was achieved by estimating the number of constrained frames between each two regularly encoded frames required to meet the complexity target. Furthermore, the same authors presented an extension of this work [11] which intended to improve the previous solution for the case of fast-motion video sequences with low target complexities. In particular, they proposed to estimate the maximum CU depth to be explored in the constrained frames based on both spatial and temporal correlations, such that spatial neighboring CUs were also used along the CUs

of previous frames to estimate the CU depths for the current constrained frame.

In [12] a rate-distortion and complexity optimization method was proposed to select the quantization and depth parameters. Using predictive techniques and game theory-based methods, the maximum CU depths for some frames were restricted in order to carry out the encoding within a given complexity budget.

In [13] a workload management scheme was suggested. The idea was to dynamically control the complexity by estimating a complexity budget for each frame. Specifically, this method acted on different parameters of the encoder, such as the maximum CU depth assessed, the search range, etc., so that the complexity target was fulfilled. In order to minimize the RD losses, a set of different encoding configurations was designed as well as a control feedback loop to dynamically update the available computational resources.

In [14] a fast CU depth decision method was described. By analyzing the CU depths selected in the last frame, the least used CU depths were disabled in the current frame. Moreover, for every CU, the depths of the neighboring and co-located CUs were analyzed to avoid checking unnecessary CU depths. The decision at frame level depended on some thresholds and the decision at CU level depended on the number of neighboring CUs that fulfilled some requirements. The thresholds and the number of neighbors were fixed experimentally.

In [15] a fast CU depth selection relying on a Bayesian approach was presented. The algorithm was based on estimating offline some class-conditional probability density functions (*pdfs*) that were stored in a look up table. Subsequently, using a Bayesian decision rule, the thresholds to decide whether to check or not the next CU depth were estimated (also offline) for different encoding settings and sequence resolutions.

[16] also presented a method able to constraint the CU depths by predicting the optimal depth from spatial-neighboring and co-located CTBs. Moreover, three early termination methods to select a suitable PU partition were also described.

[17] was based on a so-called "pyramid motion divergence" to early skip some CU depths. First, the optical flow was estimated from a down-sampled original (non-enconded) frame. Then, for each CU, the pyramid motion divergence was calculated as the variance of the optical flow of the current CU with respect to that of the CUs of smaller size. Finally, since CUs with similar pyramid motion divergence tended to use a similar splitting, an algorithm based on Euclidean distances was used to select a suitable CU quadtree structure.

In [18] a simple CU early termination method was suggested. In particular, they proposed to stop the CU splitting process when the selected PU mode for the current CU depth was the Skip mode.

In [19] a method was presented that dealt with decisions at the CU and PU levels. The PU mode decision was terminated if the R-D cost was below a threshold. This threshold was calculated as the average R-D cost of some blocks previously encoded with Skip mode. The CU depth early decision was made by comparing the R-D cost at a certain depth with the sum of the best R-D costs of the four CUs of the following depth. If the latter was higher, the CU was not split anymore.

[20] presented a CU depth decision method based on the depth correlation information between adjacent CUs in the spatial and temporal neighborhoods. The depth search range was adaptively selected for each CU based on the information of the more selected CU depths in the spatial and temporal neighboring CUs. With such information in the CU to be encoded, a similarity degree is selected with different depth search ranges available.

In [21] a fast decision algorithm for Intra prediction was described. Three early termination methods were proposed. The first one, avoided the computation of large PUs based on the correlation of neighboring PUs. The second and third methods, which rely on the Bayes' rule on RD costs, either skipped a specific PU partition and proceed to the next or skipped all the remaining PUs, respectively.

Although a lot of research has been devoted to complexity reduction in HEVC, there is still much room for improvement in the CC field. Some of the previous CC works suffered from slow convergence to the target complexity (e.g., [10] and [11]). In other cases, the complexity was controlled by means of the dynamical selection of encoding configuration parameters without taking into account the potential impact on the performance of every parameter (e.g., [13]). Other methods were not able to adapt to the video content (e.g., [14] or [15], which are actually complexity reduction methods that rely on either fixed thresholds or statistics calculated offline).

The main contribution of this work is to use a CU early termination method based on a RD cost analysis to design a CC method. To the best of our knowledge, these techniques are commonly used in complexity reduction frameworks, but they have not been applied to address the CC problem in HEVC. The proposed method relies on adaptive thresholds computed based on RD cost statistics and actual encoding time measures to control the complexity *on-the-fly*. In this way, its behavior is adapted over time to the video content, to the encoder configuration, and to variable target complexities. Finally, our method only needs to perform simple mathematical operations to update the parameters; in other words, there is no extra complexity associated with the method itself.

This work is a novel CC design for the HEVC standard inspired by an earlier work by the same authors [22] that addressed the CC problem in the H.264/AVC standard. The method in [22] focused on the mode decision in H.264/AVC, where a hypothesis test was used to choose between a low- or a high-complexity encoding mode, and the threshold of such a test was adapted according to the target complexity. While mode decision in H.264 is more related to PU selection in HEVC, the method proposed in this paper relies on CU depth decision, where it is more effective to tackle the CC problem in HEVC. Consequently, a comprehensive "ad-hoc" analysis has been conducted to understand the encoder performance regarding the CU depth decision. Moreover, since the decision has to cover all the available CU depths, we have proposed to make a split or non-split decision at every depth level, so that several thresholds (one for each possible CU depth) need to be managed to reach a given target complexity.

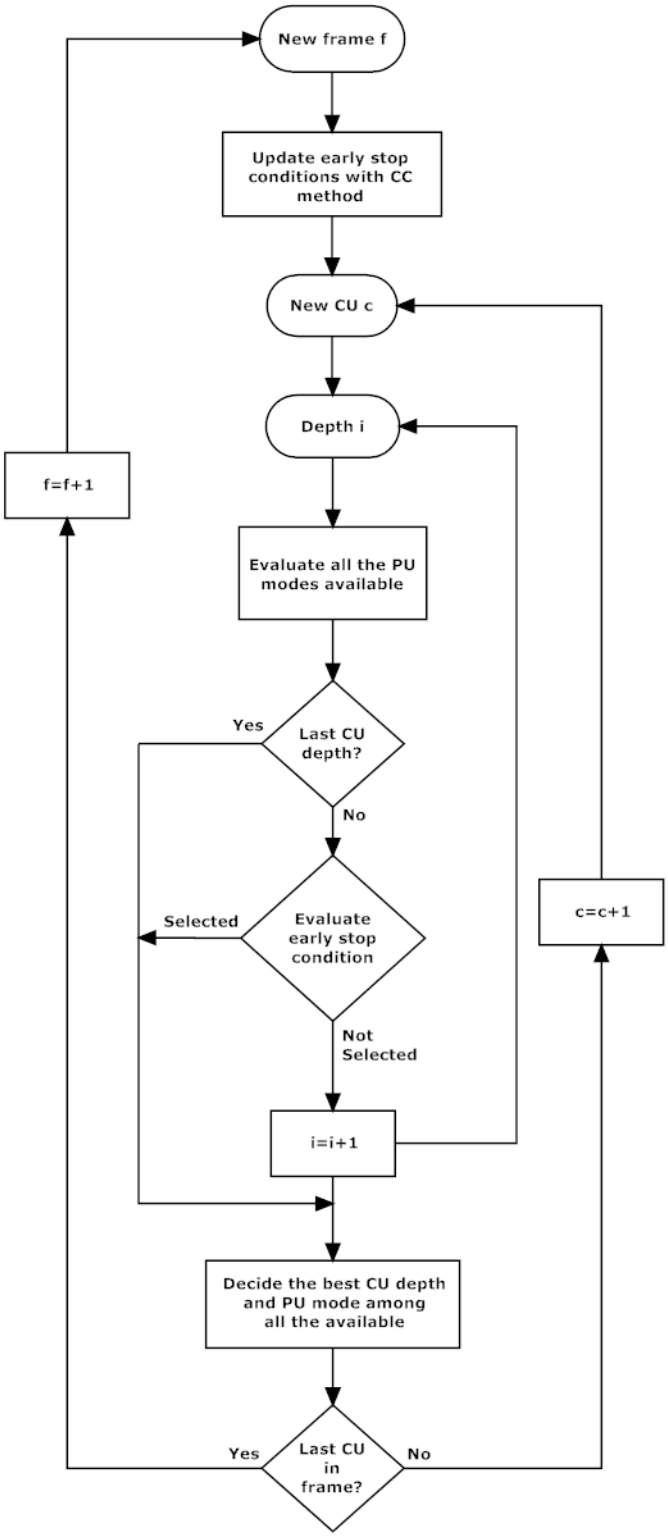


Fig. 3. Flowchart of the proposed method.

III. Proposed method

A. Overview

In order to control the computational complexity of an HEVC standard encoder, the proposed method, which only applies to inter slices, establishes an early stop condition at each CU depth based on a set of dynamically adjusted thresholds. Fig. 3 shows a flowchart that summarizes the main steps of the method. For every frame f and every CU c, the method starts exploring all the possible PU modes for the lowest depth level $i = 0$. If the early termination condition is satisfied, the higher depth levels are not tested, saving the corresponding computation time. On the contrary, if the early termination is not made, the encoder continues evaluating the next CU depth $i = i + 1$, checking again the corresponding termination condition. In summary, the goal of the proposed method is to determine a suitable number of depths to explore, so that certain complexity constraint is met and the encoder does not incur in significant coding efficiency losses. To that purpose, the proposed method has to be content-dependent, i.e., has to adapt to the actual video sequence content; moreover, the bit rate and quality must be maintained near to that obtained with the regular coding process.

To establish the early termination conditions we rely on a set of thresholds (one per depth level) that depend on statistics of R-D costs and actual time encoding measures. The R-D costs (2) help us to select an appropriate CU depth while not incurring significant coding efficiency losses. The time encoding measures allow us to adjust the encoding process to the target complexity in real-time. In other words, having real-time measures of the time spent by the encoding process, we are capable of dynamically adjusting the thresholds to make a suitable number of early stops to meet the target complexity.

In [23] we proved that the statistical distributions of the R-D costs could be used to dynamically select a threshold which allowed us to make reliable decisions concerning the mode decision problem in H.264/AVC. Therefore, we decided to explore the same idea on HEVC and, moreover, to propose a full CC algorithm. The suitability of the R-D costs for making early stops and the design of the set of thresholds and their *on-the-fly* adaptation are explained in the next subsections.

TABLE I
A priori PROBABILITIES (%) OF EVERY CU DEPTH.

		Depth 0	Depth 1	Depth 2	Depth 3
BasketballDrill	**QP 22**	0.17	0.33	0.25	0.19
	QP 27	0.33	0.29	0.21	0.11
	QP 32	0.44	0.27	0.18	0.05
	QP 37	0.54	0.16	0.13	0.03
BQMall	**QP 22**	0.16	0.25	0.30	0.23
	QP 27	0.26	0.27	0.25	0.15
	QP 32	0.35	0.29	0.22	0.09
	QP 37	0.44	0.29	0.17	0.05
FourPeople	**QP 22**	0.50	0.27	0.16	0.06
	QP 27	0.71	0.18	0.08	0.02
	QP 32	0.81	0.13	0.04	0.01
	QP 37	0.86	0.09	0.03	0.01

B. Motivation

First, we study if the R-D costs are adequate variables to rely on when designing a fast CU decision algorithm intended to control the complexity in HEVC.

The aim of our analysis is threefold: first, to study the relationship between CU depths and both the actual video content of the sequences and the quantization parameter (QP); second, to check if the R-D costs are useful to early decide

TABLE II
MEANS AND STANDARD DEVIATIONS OF THE R-D COSTS ASSOCIATED WITH EVERY PU MODE WHEN DEPTH 0 IS OPTIMAL AND WHEN IT IS NOT.

			$J_{Merge,0}$	$J_{2N\times 2N,0}$	$J_{2N\times N,0}$	$J_{N\times 2N,0}$	$J_{Min,0}$
BasketballDrill	**QP 22**	$\mu\|d^*=0$	27242	27465	27224	27237	26986
		$\sigma\|d^*=0$	4422	4742	4427	4464	4383
		$\mu\|d^*\neq 0$	59646	59136	57142	56345	54722
		$\sigma\|d^*\neq 0$	35805	34893	32894	31973	30841
	QP 27	$\mu\|d^*=0$	52231	52530	52411	52396	51713
		$\sigma\|d^*=0$	11363	11186	10967	10952	10886
		$\mu\|d^*\neq 0$	138970	134770	129340	127230	123050
		$\sigma\|d^*\neq 0$	87177	82628	77502	75642	72824
	QP 32	$\mu\|d^*=0$	98320	99380	99605	99376	96781
		$\sigma\|d^*=0$	38847	36279	35970	35895	35421
		$\mu\|d^*\neq 0$	295040	281590	268250	263810	253540
		$\sigma\|d^*\neq 0$	180690	169120	156780	153640	147240
	QP 37	$\mu\|d^*=0$	205820	208940	209950	208570	199530
		$\sigma\|d^*=0$	137410	129050	129360	128330	127230
		$\mu\|d^*\neq 0$	568130	541880	515360	506030	482480
		$\sigma\|d^*\neq 0$	338160	318710	295790	289790	277560
FourPeople	**QP 22**	$\mu\|d^*=0$	14940	15106	15088	15071	14911
		$\sigma\|d^*=0$	6668	6651	6654	6637	6644
		$\mu\|d^*\neq 0$	29571	29466	28974	28959	28489
		$\sigma\|d^*\neq 0$	15777	15674	14983	14953	14426
	QP 27	$\mu\|d^*=0$	25449	26117	26025	25970	25344
		$\sigma\|d^*=0$	15031	14940	14952	14917	14925
		$\mu\|d^*\neq 0$	62849	62140	60130	60165	58397
		$\sigma\|d^*\neq 0$	39621	39044	36588	36405	34723
	QP 32	$\mu\|d^*=0$	47860	50539	50227	50011	47545
		$\sigma\|d^*=0$	33492	33115	33232	33151	33014
		$\mu\|d^*\neq 0$	134010	131760	126510	126930	122180
		$\sigma\|d^*\neq 0$	89218	86372	80503	79659	75947
	QP 37	$\mu\|d^*=0$	100610	109720	109340	108720	99958
		$\sigma\|d^*=0$	76376	75367	75805	75896	75265
		$\mu\|d^*\neq 0$	281280	278290	267870	268910	256700
		$\sigma\|d^*\neq 0$	170070	166500	156110	154610	146940

the CU depth in HEVC; and third, to find which one of the available R-D costs turns out to be more suitable for our purpose.

First, we study the *a priori* probabilities of every CU depth in different video sequences and for different quality targets. In this manner, we intend to gain some insights into the relationships between the optimal CU depths and the contents of sequence and the QP. For this purpose, we have used the HM13.0 software [24] with the configuration file "encoder_lowdelay_P_main" (with this configuration the maximum CU size is 64×64 pixels -depth 0- , and the minimum is 8×8 pixels -depth 4-). Following the specifications in [25], a subset of the recommended test sequences were encoded with QP values of 22, 27, 32, and 37. In this conditions, we estimated the *a priori* probabilities of every CU depth considering the complete encoded sequence. For brevity reasons, we only show in Table I the results for three representative sequences since similar conclusions can be drawn from others.

As can be observed, for the sequences with smooth or little movement and static regions (such as "FourPeople") the encoder selects the lower depths with high probability for all QP values. However, the probability of selecting lower depths when the sequence is more complex decreases notably. Moreover, the *a priori* probability of depth 0 (or even the depth 1 in "BQMall") increases with QP and vice versa. This means that a coarser coding process results in a larger number of CUs encoded using large sizes (such as 64×64 or 32×32). In view of these results, which shows that low CU depths tend to be optimal for several sequences and QPs, an early determination of the optimal CU depth favoring low depths should undoubtedly contribute to reduce the complexity of the encoding process. Moreover, it seems reasonable to design an algorithm able to adapt its behavior *on-the-fly* to the content and the encoder configuration since the optimal depth selection shows high dependence on both the sequence and the QP value.

Second, we checked that the R-D costs are suitable variables to early decide the CU depth in the HEVC standard. As explained before, at every CU depth there are several PU modes that will be evaluated by the encoder in R-D terms to select the best one for that depth. We denote by $J_{PU=a,depth=i}$ the R-D cost associated with the PU mode a at the depth i. Specifically, we analyzed the statistics (means and standard deviations) of the R-D costs associated with every mode a at depth i given both that depth i was optimal and that it was not, i.e., $J_{PU=a,depth=i}|depth^* = i$ and $J_{PU=a,depth=i}|depth^* \neq i$, respectively.

The experimental setup was the same as for the previous analysis. In Table II we show the results for two sequences ("BasketballDrill" and "FourPeople") at the four recommended QP values for depth 0. In Table II, $J_{Merge,0}$ refers to the R-D cost associated with the "Merge" PU mode at depth 0, $J_{Min,0}$ refers to the minimum cost obtained after checking all the PU modes available (depth 0), $J_{2N\times 2N,0}$ refers to the cost for the 2N×2N PU mode (depth 0), and so on. $\mu|d^*=0$ and $\sigma|d^*=0$ denote the mean and the standard deviation when the optimal depth is 0, while $\mu|d^*\neq 0$ and $\sigma|d^*\neq 0$ denote these same statistics when the optimal depth is other than 0.

In order to visually check if the two conditional *pdfs* of the R-D costs, given that depth i was optimal and that it was not, are actually separable, we have estimated both *pdfs*, as follows:

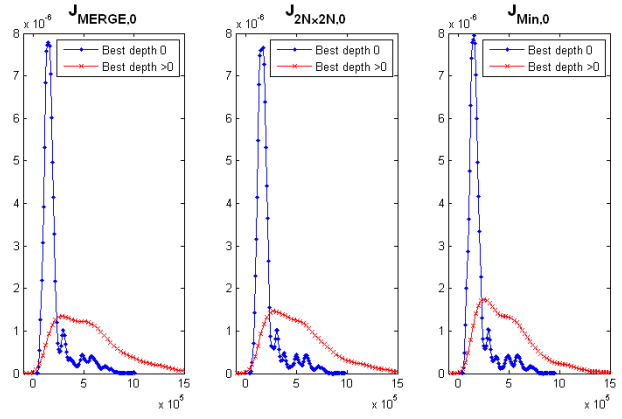
$$p_J\left(J_{PU=a,depth=i}|depth^*=i\right)$$
$$p_J\left(J_{PU=a,depth=i}|depth^*\neq i\right), \quad (3)$$

i.e., the probability of certain cost $J_{PU=a,depth=i}$ when the optimal CU depth is i, and when it is not. Fig. 4 shows the obtained *pdfs* for depth 0 (for higher depths the results are similar). In this manner, we gain some insight into the shape and the actual separability of these *pdfs*. The sequences "BasketballDrill" at QP 37 and "FourPeople" at QP 22 were used to illustrate this point.

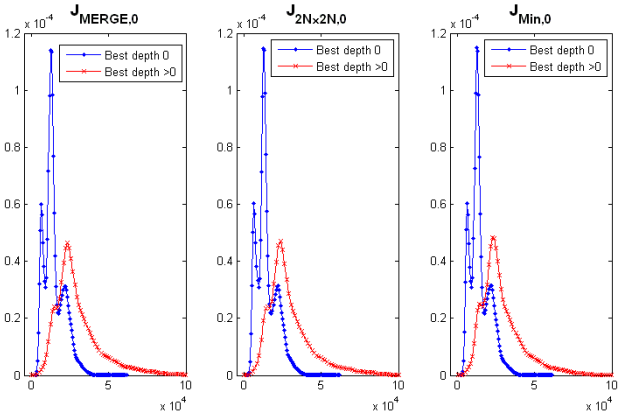
The results both in Table II and Fig. 4 show that the statistics of the two compared *pdfs* (when certain depth is optimal or not) are significantly different and reasonably separable for all the analyzed R-D costs and that all the analyzed $J_{PU=a,depth=i}$ costs present a very similar behavior. In particular, we draw two conclusions: 1) the CU depth decision problem can be addressed from updated statistical information of these R-D costs; and 2) any of the considered R-D costs turns out to be suitable. In the following subsection, we proposed a specific design of the early stops based on one of these R-D costs.

C. Designing the early termination conditions

As explained before, the proposed method aims to make fast CU depth decisions which allows us to control the complexity of the encoding process. To this purpose, the method relies on



(a) "BasketballDrill" at QP 37



(b) "FourPeople" at QP 22

Fig. 4. An illustration of the *pdfs* for several R-D costs at CU depth 0 for two sequences, "BasketballDrill" and "FourPeople".

a dynamical selection of a set of thresholds, one per depth level, based on statistical information related to R-D costs that is updated *on-the-fly*, so that the fast CU depth decision process is actually content-dependent.

In the previous section it became evident that there are relevant differences between the *pdfs* when certain depth is optimal or not (i.e., between making or not a split decision at certain depth) and that each of the resulting *pdf* pairs in (3) is separable by establishing a proper threshold. Furthermore, since these differences are consistent for different R-D costs (e.g., similar behavior can be seen in Fig. 4 for $J_{MERGE,0}$, $J_{2N\times 2N,0}$, and $J_{Min,0}$), we decided to use $J_{2N\times 2N,i}$ for convenience, because it provided slightly better results than the others.

In order to make the process simpler, we rely on just one *pdf* to make our decision, i.e., instead of seeking for an optimal threshold taking into account both the conditional *pdf* given $depth^* = i$ and the conditional *pdf* given $depth^* \neq i$, we make our decision at every depth level just considering $(p_J\left(J_{PU=a,depth=i}|depth^* = i\right))$, as we successfully proposed in [23] in the H.264/AVC framework. Specifically, for every CU depth i, we set a threshold that directly depends on the mean and the standard deviation of this *pdf*, i.e.:

$$TH_{depth=i} = \mu_{J_{2N\times 2N,i}} + \left(n_{depth=i} \times \sigma_{J_{2N\times 2N,i}}\right), \quad (4)$$

where $TH_{depth=i}$ is the threshold for depth level i; $\mu_{J_{2N\times 2N,i}}$ and $\sigma_{J_{2N\times 2N,i}}$ are the mean and standard deviation of $p_J\left(J_{2N\times 2N,i}|depth^* = i\right)$, respectively; and $n_{depth=i}$ is the control parameter, which allows us to adapt the threshold. Defining the threshold in this manner provides two essential advantages: 1) since the mean and standard deviation are updated on a CU-by-CU basis (details below), the thresholds are content-adaptive; and 2) the $n_{depth=i}$ parameter allows us to set more or less demanding thresholds according to the target complexity.

For each CU at depth level i, if the actual cost $J_{2N\times 2N,i}$ is below the threshold, the early termination condition is satisfied, and the process to encode that CU is stopped. On the other hand, if $J_{2N\times 2N,i}$ is above the threshold, the encoding process for that CU continues exploring higher depth levels. In other words, the early stop at a specific depth level i is actually made if the likelihood of the actual cost $J_{2N\times 2N,i}$ coming from the conditional *pdf* given that $depth^* = i$ is high enough (relative to the threshold).

Finally, the parameter $n_{depth=i}$ establishes a balance of the *complexity-coding efficiency* tradeoff for a given sequence, encoding configuration, and complexity target. In particular, the control of this tradeoff allows us to act on how often the early stop at i is actually made and, consequently, allows us to dynamically control the complexity. More details concerning how to manage this parameter are given in Section III-D.

1) On-the-fly estimation of the statistics: The fact that the thresholds that define the early termination conditions are content-adaptive is one of the main contributions of this paper. In this subsection we describe how the *pdf* parameters are estimated *on-the-fly* to adapt the threshold to the content along the video sequence.

In particular, we model the *pdfs* from their means and variances. Thus, we suggest a simple procedure to update these two parameters on a CU-by-CU basis which is based on an exponential average:

$$\hat{\mu}_{J_{2N\times 2N,i}}(t) = \alpha\hat{\mu}_{J_{2N\times 2N,i}}(t-1) + (1-\alpha)J_{2N\times 2N,i}(t), \quad (5)$$

$$\hat{\sigma}^2_{J_{2N\times 2N,i}}(t) = \alpha\hat{\sigma}^2_{J_{2N\times 2N,i}}(t-1) + (1-\alpha)(J_{2N\times 2N,i}(t) - \hat{\mu}_{J_{2N\times 2N,i}}(t))^2, \quad (6)$$

where t is an index associated with the times that the depth level i is selected as optimal; $\hat{\mu}_{J_{2N\times 2N,i}}(t-1)$ and $\hat{\mu}_{J_{2N\times 2N,i}}(t)$ are the estimated means at the instants $(t-1)$ and t, respectively (the variances $\hat{\sigma}^2_{J_{2N\times 2N,i}}(t-1)$ and $\hat{\sigma}^2_{J_{2N\times 2N,i}}(t)$ following the same notation); $J_{2N\times 2N,i}(t)$ is the RD cost at the instant t; and α is the parameter defining the forgetting factor of the exponentially averaging process. Specifically, α has been set to 0.95 in our experiments.

Fig. 5 provides an example illustrating the behavior of the exponential average. In particular, we show the estimated mean $(\hat{\mu}_{J_{2N\times 2N,i}})$ along 350 samples. Furthermore, the expected mean values considering blocks of 50 samples are shown, as control points that the average estimator should follow. As can be observed, the exponential average produces a reasonable approximation. The behavior of the estimation of the variances is nearly identical.

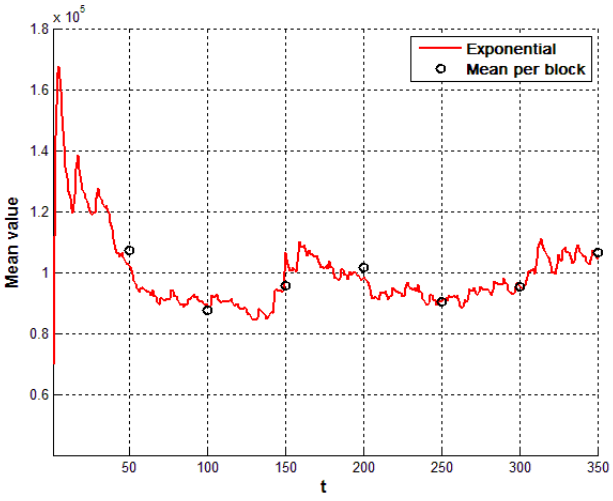


Fig. 5. Illustration for the exponentially average estimation of $\hat{\mu}_{J_{2N\times 2N,i}}$ for BasketballDrill at QP 32.

D. Controlling the complexity

As stated before, $n_{depth=i}$ controls the number of early stops selected at depth level i. In particular, from (4), the higher $n_{depth=i}$, the higher the resulting threshold and the more likely the early stop (since the likelihood of obtaining a $J_{2N\times 2N,i}$ cost lower than $TH_{depth=i}$ becomes higher). Therefore, acting on $n_{depth=i}$, we will be able to manage the computational complexity to meet certain target. To be more precise, hereafter we will use encoding times as practical measures of the computational complexity.

Specifically, following a strategy similar to that we proposed for H.264/AVC [22], we rely on a simple feedback algorithm for updating $n_{depth=i}$:

$$n_{depth=i}^{fr=j+1} = n_{depth=i}^{fr=j} + \lambda_{depth=i}\left(time^{fr=j} - time_{target}\right), \tag{7}$$

where $fr = j+1$ and $fr = j$ stand for frame numbers $j+1$ and j, respectively; thus, $n_{depth=i}^{fr=j+1}$ varies from its previous value (that of frame j) according to the deviation of the actual encoding time of frame j with respect to the target, i.e., $\left(time^{fr=j} - time_{target}\right)$. Furthermore, the parameter $\lambda_{depth=i}$ controls the tradeoff between the speed and the accuracy of the algorithm.

In a few words, $n_{depth=i}$ is updated *on-the-fly* on a frame-by-frame basis to meet the complexity constraint. For example, if $time^{fr=j}$ is higher than $time_{target}$, it means that we have allocated to frame j more resources than those targeted and, thus, we need to increase the threshold (to make more early stops), as it really happens since $time^{fr=j} - time_{target}$ becomes positive.

It should be noticed that the parameter $\lambda_{depth=i}$ must depend on the depth level i. In doing so, we aim to produce smoother transitions of $n_{depth=i}$ when i is low, which is desirable because of, if we make a wrong decision by making an early stop at a low depth, we will incur in significant coding efficiency losses. Thus, we only allow smooth transitions of $n_{depth=i}$ for low depths and relax this constraint as the depth increases. In particular, $\lambda_{depth=i}$ is computed as follows:

$$\lambda_{depth=i} = \lambda_0 \times \left(time_{depth=i}/time_{CU}\right), \tag{8}$$

where λ_0 is an initial value experimentally set to 0.2; $time_{depth=i}$ is the time to encode a CU with an early stop at depth i; and $time_{CU}$ is the time to encode a CU exploring all depths available. As can be easily noticed, $time_{CU}$ is the maximum value for $time_{depth=i}$ and $time_{depth=i+1} > time_{depth=i}$. Therefore, the lower the depth i, the lower $\lambda_{depth=i}$ and vice-versa. It should be noted that $time_{depth=i}$ and $time_{CU}$ are computed in a frame-by-frame basis as the average time spent encoding a CU with an early stop at depth i and the average time devoted to each CU when all depths are explored, respectively.

The target complexity can be specified by the user or the application running the video encoder, and depends on the resources available in a specific device and the time that the user/applications allocates for the encoding process. In the proposed method the target complexity is specified as a percentage with respect to the complexity of a full-search encoding process $TC(\%)$, and our method is responsible for transforming this percentage in an encoding target time value $time_{target}$ to be used in (7). Specifically, knowing the time that a CU needs to be encoded using full-search $(time_{CU})$, and the total number of CUs per frame M, we can easily obtain $time_{target}$ as follows:

$$time_{target} = time_{CU} \times M \times (TC/100). \tag{9}$$

E. Summary of the algorithm

The complete method is summarized in Algorithm 1.

Algorithm 1 Proposed coding process.

Require: F: number of frames to be encoded
Require: M: number of CUs per frame
Require: D: number of available depths in a CU
1: **for** $\forall f \in F$ **do**
2: Retrieve the time to encode the previous frame $time^{fr=f-1}$
3: Obtain average values $time_{CU}$ and $time_{depth=d}$
4: Calculate $time_{target}$ to encode the frame f (9)
5: Calculate $\lambda_{depth=d}$ (8) and $n_{depth=i}$ (7)
6: **for** $\forall m \in M$ **do**
7: **for** $\forall d \in D$ **do**
8: Evaluate all PU partition modes in depth d
9: Calculate the $\hat{\mu}_{J_{2N\times 2N,d}}$ (5) and $\hat{\sigma}^2_{J_{2N\times 2N,d}}$ (6)
10: Obtain $TH_{depth=d}$ (4)
11: **if** $J_{2N\times 2N,d} < TH_{depth=d}$ **then**
12: Go to 14
13: **end if**
14: **end for**
15: Decide the best coding options among all CU depths and PU partitions evaluated
16: **end for**
17: **end for**

TABLE III
PERFORMANCE EVALUATION OF THE PROPOSED METHOD FOR DIFFERENT TARGET COMPLEXITIES.

TC	$TC = 90\%$			$TC = 80\%$			$TC = 70\%$			$TC = 60\%$		
Sequence	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$
BasketballDrill (832×480)	0.67	0.02	9.57	0.79	0.02	17.06	1.23	0.05	28.44	5.19	0.24	39.76
BasketballDrillText (832×480)	0.48	0.02	8.52	0.75	0.03	17.28	1.11	0.05	27.75	6.35	0.30	41.12
BasketballDrive (1920×1080)	0.07	0.00	9.30	0.32	0.01	17.10	0.83	0.02	26.51	1.72	0.05	38.21
BasketballPass (416×240)	0.42	0.02	13.68	1.11	0.06	20.44	2.10	0.11	27.82	4.84	0.25	35.15
BlowingBubbles (416×240)	0.90	0.02	12.62	2.26	0.08	20.43	5.74	0.24	32.95	11.79	0.47	43.75
BQMall (832×480)	0.31	0.01	12.42	0.71	0.03	20.78	2.42	0.11	31.44	12.20	0.49	46.59
BQSquare (416×240)	0.54	0.01	11.56	1.40	0.05	17.75	3.74	0.15	26.50	12.74	0.58	39.14
BQTerrace (1920×1080)	0.06	0.00	9.98	0.59	0.02	20.60	1.77	0.06	33.65	7.92	0.25	54.10
Cactus (1920×1080)	0.10	0.00	7.19	0.14	0.01	14.91	0.74	0.03	25.22	2.42	0.08	39.58
ChinaSpeed (1024×768)	0.17	0.01	12.73	0.89	0.07	21.15	4.03	0.29	30.58	26.67	1.38	42.66
FourPeople (1280×720)	−0.01	0.00	12.55	0.16	0.01	20.93	0.35	0.01	31.57	0.66	0.02	44.46
Johnny (1280×720)	−0.04	0.00	9.74	0.08	0.00	19.35	−0.06	0.00	28.84	−0.10	0.00	39.44
Kimono (1920×1080)	0.04	0.00	8.14	0.29	0.01	19.69	0.72	0.03	30.25	1.07	0.04	42.06
KristenAndSara (1280×720)	0.12	0.00	9.46	−0.18	0.00	18.39	0.07	0.00	28.08	0.12	0.00	39.08
ParkScene (1920×1080)	0.03	0.00	9.65	0.57	0.03	19.51	1.57	0.07	31.14	7.28	0.31	48.62
PartyScene (832×480)	1.13	0.05	9.97	3.62	0.17	21.18	11.40	0.55	37.91	25.05	1.02	52.67
RaceHorses (416×240)	0.12	0.01	5.60	0.92	0.05	13.77	3.82	0.21	24.17	15.42	0.78	38.92
SlideEditing (1280×720)	0.31	0.04	8.78	0.23	0.02	17.66	0.93	0.13	28.96	0.48	0.07	42.49
SlideShow (1280×720)	−0.22	−0.01	7.06	0.50	0.04	16.58	1.61	0.14	25.16	6.97	0.54	35.56
Vidyo1 (1280×720)	0.12	0.00	9.75	0.24	0.01	18.61	0.14	0.00	28.80	0.26	0.01	39.48
Vidyo3 (1280×720)	−0.05	0.00	7.16	−0.17	0.00	15.93	−0.08	0.00	26.74	0.46	0.01	41.46
Vidyo4 (1280×720)	−0.02	0.00	10.13	0.01	0.00	18.07	0.29	0.01	29.32	0.87	0.04	44.83
Average	**0.23**	**0.01**	**9.79**	**0.69**	**0.03**	**18.50**	**2.02**	**0.10**	**29.17**	**6.83**	**0.31**	**42.23**

IV. EXPERIMENTS AND RESULTS

A. Experimental setup

To assess the performance of the proposed method, it was implemented in HM13.0 software [24]. The test conditions were chosen following the recommendations in [25] and the configuration file was "encoder_lowdelay_P_main". A comprehensive set of sequences of several resolutions and covering a variety of video contents (motion, textures, etc.) was used (see Table III).

B. Meeting a target complexity

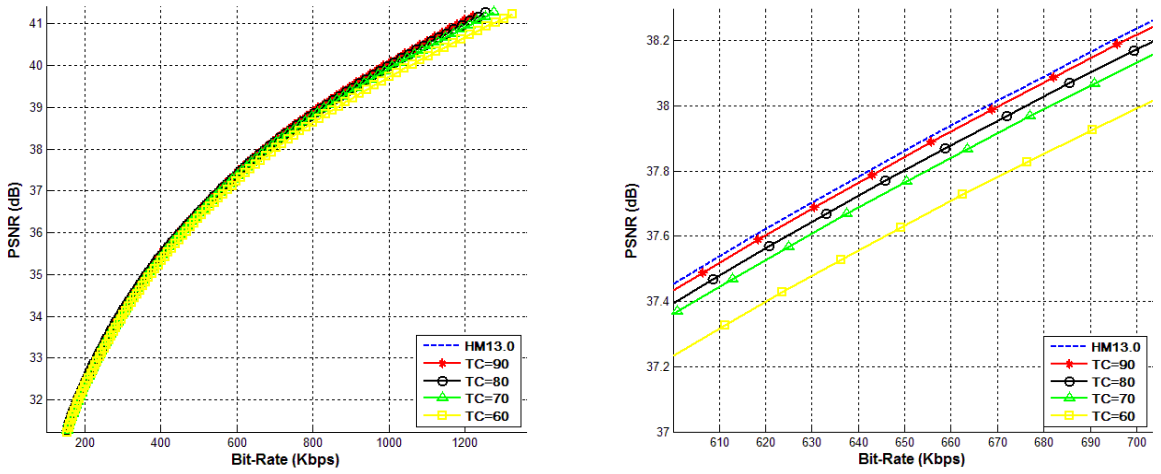
To compute the encoding time savings and, consequently, the complexity reductions, we have used the following measure:

$$TS = \frac{Time(HM13.0) - Time(Proposed)}{Time(HM13.0)} \times 100. \quad (10)$$

Thus, relying on this measure we were able to assess the capability of our proposal to meet different target complexities. Moreover, as another goal in our proposal is to incur in coding efficiency losses as low as possible, we also needed to monitor the performance in terms of bit rate increment ΔBR or PSNR loss $\Delta PSNR$, which were calculated following the recommendations in [26]. In particular, the proposed method was evaluated for four different target complexities, $TC(\%) = \{90, 80, 70, 60\}$.

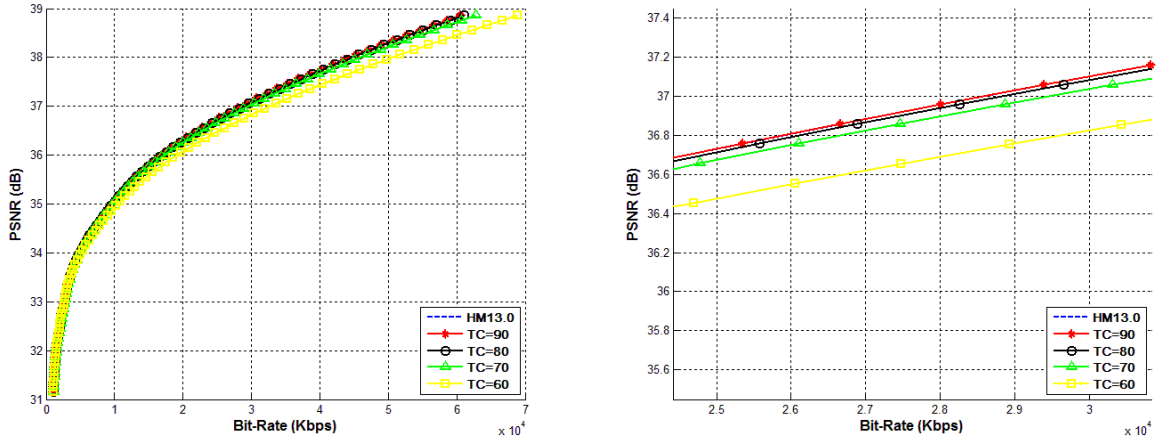
Table III shows the results for all the considered target complexities in terms of ΔBR, $\Delta PSNR$, and TS, averaged across the four QP values recommended in [25] (QP values of 22, 27, 32, and 37). From these results, we can conclude that the proposed method obtains a notable accuracy for all the considered target complexities. In particular, the mean values of TS obtained taking into account all the video sequences show a very small deviation from the target values (the highest deviation is 2.23% for TC=60%). Moreover, the complexity savings are obtained in exchange for very limited losses in coding performance. The mean values for ΔBR are 0.23, 0.69 and 2.2% for TCs of 90, 80, and 70%, respectively. However, when TC is 60%, the average ΔBR reaches 6.8%, which was expected since the proposed only acts on the CU depth, and, necessarily, low target complexities involve large CU sizes and, therefore, a coarser encoding process.

Furthermore, it should be noticed that the average 6.8% in ΔBR comes from few sequences for which achieving 40% complexity reduction is hard (only 6 sequences out of 22 produce BD-Rate loss above 10%, causing the average value



(a) R-D performance. (b) Zoom in of the Figure on the left.

Fig. 6. R-D performance for the 4 considered target complexities for the sequence *BasketballPass*.



(a) R-D performance. (b) Zoom in of the Figure on the left.

Fig. 7. R-D performance for the 4 considered target complexities for the sequence *BQTerrace*.

to reach 6.83%).

Some selected numerical results of Table III have been further illustrated on Figs. 6 and 7, which show the R-D performance of the proposed method for different target complexities. On the left part of the figures the complete R-D curves are shown, while on the right part we have zoomed in the curves to see a segment in more detail. As can be observed, the conclusions do not change with respect to those already drawn from Table III.

C. Comparison with a complexity control state-of-the-art method

In this subsection we compare the proposed method with a state-of-the-art method [11]. The configuration file and QPs used to conduct this comparison are those of the previous experiment, and the comparison is focused on target complexities $TC = \{80, 70\}$ because the method described in [11] is not able to reach $TC = 60\%$. The obtained results are shown in Table IV in terms of ΔBR, $\Delta PSNR$, and TS, averaged across the four QP values.

As can be seen, the method presented in [11] is not as accurate in terms of meeting the target complexity as the proposed method when the target complexity becomes more demanding. Nevertheless, in general, both methods achieve a high accuracy level. In particular, for $TC = 80\%$ the deviation from the target is below 2% in both cases; for $TC = 70\%$ the deviation of the proposed method continues below 2% while in the case of [11] it increases up to 3.6%. Furthermore, for $TC = 70\%$ and some specific sequences such as *Johnny*, *BasketballPass*, or *BlowingBubbles*, the proposed method obtains significantly higher time savings (28.84, 27.82, and 32.95 %, respectively) than those of [11] (21.55, 21.39, and 21.11%).

Regarding losses in coding efficiency, the average ΔBR of [11] is higher than that of the proposed method; specifically, the results of [11] are about 3% worse than those of the

TABLE IV
PERFORMANCE EVALUATION OF THE PROPOSED METHOD IN COMPARISON WITH [11].

	Proposed method						[11]					
TC	$TC = 80\%$			$TC = 70\%$			$TC = 80\%$			$TC = 70\%$		
Sequence	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$
BasketballDrill (832×480)	0.79	0.02	17.06	1.23	0.05	28.44	4.45	0.18	19.52	6.99	0.27	25.60
BasketballDrillText (832×480)	0.75	0.03	17.28	1.11	0.05	27.75	4.17	0.19	18.34	6.54	0.28	25.51
BasketballDrive (1920×1080)	0.32	0.01	17.10	0.83	0.02	26.51	2.30	0.04	20.19	5.00	0.08	29.98
BasketballPass (416×240)	1.11	0.06	20.44	2.10	0.11	27.82	6.27	0.32	17.30	7.67	0.37	21.39
BlowingBubbles (416×240)	2.26	0.08	20.43	5.74	0.24	32.95	2.69	0.10	16.88	3.80	0.14	21.11
BQMall (832×480)	0.71	0.03	20.78	2.42	0.11	31.44	4.62	0.19	19.29	7.78	0.32	23.26
BQSquare (416×240)	1.40	0.05	17.75	3.74	0.15	26.50	3.09	0.12	13.67	5.79	0.19	21.73
BQTerrace (1920×1080)	0.59	0.02	20.60	1.77	0.06	33.65	1.36	0.03	18.48	2.84	0.05	25.58
Cactus (1920×1080)	0.14	0.01	14.91	0.74	0.03	25.22	1.85	0.04	21.57	3.20	0.06	25.95
ChinaSpeed (1024×768)	0.89	0.07	21.15	4.03	0.29	30.58	5.76	0.34	19.66	9.11	0.50	27.77
FourPeople (1280×720)	0.16	0.01	20.93	0.35	0.01	31.57	1.73	0.05	22.53	3.32	0.08	31.15
Johnny (1280×720)	0.08	0.00	19.35	−0.06	0.00	28.84	1.82	0.02	19.98	4.01	0.05	21.55
Kimono (1920×1080)	0.29	0.01	19.69	0.72	0.03	30.25	0.29	0.01	22.75	0.56	0.01	29.96
KristenAndSara (1280×720)	−0.18	0.00	18.39	0.07	0.00	28.08	1.55	0.04	21.49	2.09	0.04	34.63
ParkScene (1920×1080)	0.57	0.03	19.51	1.57	0.07	31.14	2.27	0.08	21.60	3.38	0.11	27.24
PartyScene (832×480)	3.62	0.17	21.18	11.40	0.55	37.91	4.18	0.18	18.78	6.27	0.27	24.41
RaceHorses (416×240)	0.92	0.05	13.77	3.82	0.21	24.17	5.30	0.29	15.72	7.41	0.39	20.22
SlideEditing (1280×720)	0.23	0.02	17.66	0.93	0.13	28.96	5.14	0.83	16.99	8.76	1.25	25.64
SlideShow (1280×720)	0.50	0.04	16.58	1.61	0.14	25.16	5.72	0.42	25.42	6.68	0.47	28.53
Vidyo1 (1280×720)	0.24	0.01	18.61	0.14	0.00	28.80	1.74	0.04	20.58	2.33	0.04	31.80
Vidyo3 (1280×720)	−0.17	0.00	15.93	−0.08	0.00	26.74	0.96	0.03	18.55	2.04	0.05	30.92
Vidyo4 (1280×720)	0.01	0.00	18.07	0.29	0.01	29.32	1.09	0.03	17.80	6.10	0.10	25.39
Average	**0.69**	**0.03**	**18.50**	**2.02**	**0.10**	**29.17**	**3.10**	**0.16**	**19.41**	**5.07**	**0.23**	**26.33**

proposed method.

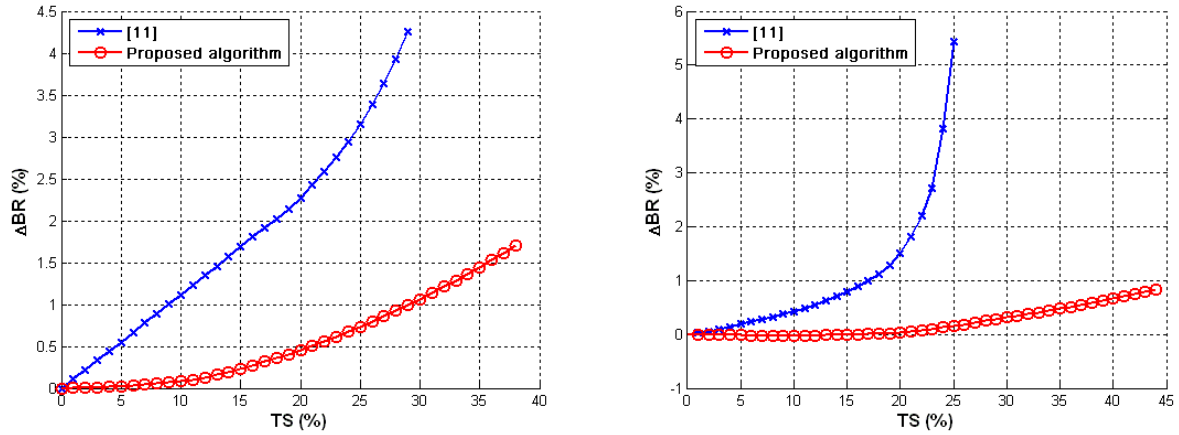
As can be inferred from results in Table IV, the method presented in [11] has more difficulties when trying to reach lower target complexities. In fact, the comparison does not involve $TC = 60\%$ because it is difficult to achieve higher TSs using [11]. This is due to the fact that this method is not able to reach the high number of constrained frames required to achieve low target complexities. Moreover, if the sequence presents high motion content, the optimal CU depths stored to be used in the constrained frames are probably high depths and, consequently, the potential complexity savings are limited. In contrast, the proposed method does not suffer from this problem since it always can use higher thresholds to reach any required target complexity.

In Fig. 8 we present ΔBR as a function of TS for the two compared algorithms with respect to the reference software. Specifically, we show the results obtained for two representative sequences *BasketballDrive* (on the left part of the figure) and *Vidyo4* (right part). As can be seen, the method in [11] is not able to achieve TSs above 30%, while the proposed method can manage TSs near to 40%. In terms of ΔBR, our proposal is able to achieve the same target complexities in exchange for notable smaller losses in coding efficiency, as can be seen in Fig. 8 for both sequences.

D. *Comparison with a complexity reduction state-of-the-art method*

In this subsection we compare our proposal with a state-of-the-art method of complexity reduction [20], which aims to reduce the complexity of the HEVC encoder, but it is not capable of meeting a given complexity target. The configuration file and QPs used to conduct this comparison are those of the previous experiments. The obtained ΔBR, $\Delta PSNR$, and TS values, averaged across the four QP values, are shown in Table V. To fairly compare both methods, we provide in Table V those results of the proposed method which are more similar to those of [20] in TS terms.

Some interesting conclusions can be extracted from this experiment. First, the method presented in [20] is not a CC method and, consequently, provides very different results in terms of TS depending on the content (e.g., in "KristenAndSara" the TS is 54%, while in "BlowingBubbles" is 8%). Therefore, it does not provide a solution capable of running



(a) *BasketballDrive*. (b) *Vidyo4*.

Fig. 8. Performance evaluation of the proposed algorithm in comparison with [11]. Bit rate increment as a function of the computational time saving.

TABLE V
PERFORMANCE EVALUATION OF THE PROPOSED METHOD IN COMPARISON WITH [20].

	Proposed method			[20]		
Sequence	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$	$\Delta BR(\%)$	$\Delta PSNR(dB)$	$TS(\%)$
BasketballDrill (832×480)	0.79	0.02	17.06	0.97	0.02	17.96
BasketballDrillText (832×480)	0.75	0.03	17.28	1.21	0.02	18.11
BasketballDrive (1920×1080)	0.83	0.02	26.51	0.97	0.01	26.79
BasketballPass (416×240)	0.42	0.02	13.68	0.48	0.01	10.06
BlowingBubbles (416×240)	0.90	0.02	12.62	0.03	0.00	8.49
BQMall (832×480)	0.71	0.03	20.78	0.56	0.01	17.12
BQSquare (416×240)	0.54	0.01	11.56	0.12	0.00	8.78
BQTerrace (1920×1080)	1.77	0.06	33.65	0.95	0.01	28.76
Cactus (1920×1080)	0.74	0.03	25.22	0.93	0.01	29.29
ChinaSpeed (1024×768)	0.89	0.07	21.15	0.18	0.01	19.68
FourPeople (1280×720)	0.66	0.02	44.46	7.86	0.11	46.58
Johnny (1280×720)	−0.10	0.00	39.44	4.20	0.05	46.96
Kimono (1920×1080)	0.29	0.01	19.69	0.42	0.01	24.47
KristenAndSara (1280×720)	0.12	0.00	39.08	14.28	0.25	54.15
ParkScene (1920×1080)	0.57	0.03	19.51	1.31	0.02	24.89
PartyScene (832×480)	1.13	0.05	9.97	0.13	0.00	12.66
RaceHorses (416×240)	0.12	0.01	5.60	0.07	0.00	7.78
SlideEditing (1280×720)	0.48	0.07	42.49	53.95	2.66	64.81
SlideShow (1280×720)	6.97	0.54	35.56	42.66	2.21	51.30
Vidyo1 (1280×720)	0.26	0.01	39.48	9.97	0.17	49.21
Vidyo3 (1280×720)	0.46	0.01	41.46	3.05	0.04	42.94
Vidyo4 (1280×720)	0.87	0.04	44.83	6.55	0.07	42.78
Average	**0.91**	**0.05**	**26.41**	**6.85**	**0.25**	**29.70**

on a fixed-resource platform, while our approach is perfectly capable as already proved in Table III.

Second, the results obtained in "SlideEditing" and "SlideShow" deserve a comment. In these two sequences the content is quite uniform until a sudden change happens. This is hard to manage for the algorithm of [20] that suffers when a scene change occurs because only few depths are actually allowed for such frames, incurring in huge ΔBRs (which approximately doubles the original rate). It should be noticed that our method does not suffer from this problem due to the adaptive statistics that define our early detection thresholds.

Third, in general terms, the results of our proposal are better when compared at the same complexity saving level with those of [20] (e.g., "BasketballDrillText" and "Vidyo3"). In some sequences, the results of our method are slightly worse than those of [20] (e.g., "RaceHorses"). However, in average, our proposal outperforms [20], since a similar complexity saving is obtained at the expense of a clearly lower ΔBR (0.91% vs. 6.85%).

E. Convergence properties

Finally, we have studied the convergence properties of our algorithm since, from our point of view, the capability of reaching the required target complexity with certain accuracy level and in the shortest time possible becomes a relevant performance indicator. For this reason, we designed a method that is able to adapt to the content *on-the-fly*. In particular, in this section we prove that our method is able to adapt its behavior *on-the-fly* over the coding process to any kind of content, coding configuration, or complexity target.

The convergence properties of our algorithm are shown in Tables VI and VII. The average time actually spent on encoding the frames is denoted as $\widehat{time}^{fr}$, and the average target time is called $\widehat{time}_{target}$. All the results are measured in seconds. The results in Table VI are obtained with a QP value of 32, and in Table VII the QP used is 27.

Table VI shows the results for three different sequences (*Four People*, *BQMall*, and *Blowing Bubbles*), exhibiting very different contents, each one encoded with four target complexities, $TC(\%) = \{90, 80, 70, 60\}$ (the TC values are specified on the left side of the table). As it can be seen, the proposed algorithm achieves an encoding time very similar to the target. Specifically, the highest deviation from the target is only 0.63 seconds, so we can conclude that we obtain a very good accuracy with our proposal.

In Table VII we show some results that illustrates the behavior of the proposed method when the target complexity

TABLE VI
CONVERGENCE PERFORMANCE EVALUATION OF THE PROPOSED ALGORITHM.

		FourPeople (1280×720)	BQMall (832×480)	BlowingBubbles (416×240)
90	$\widehat{time}_{target}$	30.50	22.41	6.48
	$\widehat{time}^{fr}$	30.20	22.09	5.85
80	$\widehat{time}_{target}$	28.35	20.28	5.13
	$\widehat{time}^{fr}$	28.04	20.11	4.90
70	$\widehat{time}_{target}$	24.49	16.51	3.85
	$\widehat{time}^{fr}$	24.32	16.44	3.78
60	$\widehat{time}_{target}$	19.95	12.88	3.46
	$\widehat{time}^{fr}$	19.86	12.98	3.48

TABLE VII
CONVERGENCE PERFORMANCE EVALUATION OF THE PROPOSED ALGORITHM.

		FourPeople (1280×720)		BQMall (832×480)		BlowingBubbles (416×240)	
		Fr 1 - 49	Fr 50 - 100	Fr 1 - 49	Fr 50 - 100	Fr 1 - 49	Fr 50 - 100
60 - 80	$\widehat{time}_{target}$	21.72	32.37	14.89	24.59	4.24	5.13
	$\widehat{time}^{fr}$	21.70	31.97	15.41	23.96	4.81	5.57
90 - 70	$\widehat{time}_{target}$	32.09	28.13	23.60	23.28	7.22	6.22
	$\widehat{time}^{fr}$	31.50	28.51	23.64	23.25	7.24	6.04

varies over the coding process. The $\widehat{time}^{fr}$ and $\widehat{time}_{target}$ average values are obtained for the same three sequences in two cases: first, with $TC = 60\%$ from frames 0 to 49 and $TC = 80\%$ from frames 50 to 100 and, second, with $TC = 90\%$ over the first fifty frames and $TC = 70\%$ over the remaining frames (each set of frames is specified in the table as "Fr 1 - 49" and "Fr 50 - 100"). As can be observed in these examples, when a change of the target complexity happens, the proposed algorithm is able to adapt its behavior and reach the new target complexity. It must be noted that, analyzing the behavior in the frames where the change in TC happens, our method is able to quickly adapt and reach the new target complexity in a few frames.

V. CONCLUSIONS AND FURTHER WORK

In this work, we have proposed a complexity control method for the HEVC standard. The proposed method is based on a set of early termination conditions at CU depth level. The early termination conditions rely on a set of thresholds which are adjusted dynamically. The thresholds apply to R-D costs whose statistical properties are estimated *on-the-fly*, allowing the proposed method to adapt to different contents, encoder configurations and target complexity requirements which can vary over the coding process.

Our proposal has been extensively tested and the experimental results prove that the method works effectively for a great variety of sequences and complexity requirements, outperforming the results achieved by a complexity control state-of-the-art method in terms of both accuracy to reach the target complexity and coding efficiency, and showing the advantages of the complexity control approach with respect to the more common complexity reduction methods. Moreover, we have shown that the proposed algorithm is able to adapt its behavior when the complexity requirements change over time.

An interesting future work would be to extend our design to act also at the PU and TU decision levels, what undoubtedly would provide additional computational savings. Moreover, we also suggest to explore the use of more complex classifiers to obtain higher accuracy in the fast decision process.

REFERENCES

[1] J. Vanne, M. Viitanen, T. Hamalainen, and A. Hallapuro, "Comparative Rate-Distortion-Complexity Analysis of HEVC and AVC Video Codecs," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1885–1898, Dec 2012.

[2] G. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1649–1668, Dec 2012.

[3] H. Everett, "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," *Operations Research*, vol. 11, no. 3, pp. 399–417, May - Jun. 1963.

[4] J. Kim, J. Yang, K. Won, and B. Jeon, "Early determination of mode decision for HEVC," in *Picture Coding Symposium (PCS), 2012*, May 2012, pp. 449–452.

[5] T. Zhao, Z. Wang, and S. Kwong, "Flexible Mode Selection and Complexity Allocation in High Efficiency Video Coding," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 7, no. 6, pp. 1135–1144, Dec 2013.

[6] R. H. Gweon, Y.-L. Lee, and J. Lim, "JCTVC-F045 Early termination of CU encoding to reduce HEVC complexity," Fifth meeting of the Joint Collaborative Team on Video Coding (JCT-VC) Torino, IT, 2011.

[7] S. Blasi, E. Peixoto, and E. Izquierdo, "Enhanced inter-prediction using Merge Prediction Transformation in the HEVC codec," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, May 2013, pp. 1709–1713.

[8] K. Choi and E. Jang, "Early TU decision method for fast video encoding in high efficiency video coding," *Electronics Letters*, vol. 48, no. 12, pp. 689–691, June 2012.

[9] S.-W. Teng, H.-M. Hang, and Y.-F. Chen, "Fast mode decision algorithm for Residual Quadtree coding in HEVC," in *Visual Communications and Image Processing (VCIP), 2011 IEEE*, Nov 2011, pp. 1–4.

[10] G. Correa, P. Assuncao, L. Agostini, and L. da Silva Cruz, "Complexity control of high efficiency video encoders for power-constrained devices," *Consumer Electronics, IEEE Transactions on*, vol. 57, no. 4, pp. 1866–1874, November 2011.
[11] G. Correa, P. Assuncao, L. Agostini, and L. Cruz, "Coding Tree Depth Estimation for Complexity Reduction of HEVC," in *Data Compression Conference (DCC), 2013*, March 2013, pp. 43–52.
[12] A. Ukhanova, S. Milani, and S. Forchhammer, "Game-theoretic rate-distortion-complexity optimization for HEVC," in *Image Processing (ICIP), 2013 20th IEEE International Conference on*, Sept 2013, pp. 1995–1999.
[13] M. Grellert, M. Shafique, M. Karim Khan, L. Agostini, J. Mattos, and J. Henkel, "An adaptive workload management scheme for HEVC encoding," in *Image Processing (ICIP), 2013 20th IEEE International Conference on*, Sept 2013, pp. 1850–1854.
[14] J. Leng, L. Sun, T. Ikenaga, and S. Sakaida, "Content Based Hierarchical Fast Coding Unit Decision Algorithm for HEVC," in *Multimedia and Signal Processing (CMSP), 2011 International Conference on*, vol. 1, May 2011, pp. 56–59.
[15] X. Shen, L. Yu, and J. Chen, "Fast coding unit size selection for HEVC based on Bayesian decision rule," in *Picture Coding Symposium (PCS), 2012*, May 2012, pp. 453–456.
[16] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An Effective CU Size Decision Method for HEVC Encoders," *Multimedia, IEEE Transactions on*, vol. 15, no. 2, pp. 465–470, Feb 2013.
[17] J. Xiong, H. Li, Q. Wu, and F. Meng, "A Fast HEVC Inter CU Selection Method Based on Pyramid Motion Divergence," *Multimedia, IEEE Transactions on*, vol. 16, no. 2, pp. 559–564, Feb 2014.
[18] K. Choi, H.-M. Park, and E. S. Jang, "JCTVC-F092 Coding tree pruning based CU early termination," Fifth meeting of the Joint Collaborative Team on Video Coding (JCT-VC) Torino, IT, 2011.
[19] H. L. Tan, F. Liu, Y. H. Tan, and C. Yeo, "On fast coding tree block and mode decision for high-Efficiency Video Coding (HEVC)," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, March 2012, pp. 825–828.
[20] Y. Zhang, H. Wang, and Z. Li, "Fast coding unit depth decision algorithm for interframe coding in hevc," in *Data Compression Conference (DCC), 2013*, March 2013, pp. 53–62.
[21] K. Lim, J. Lee, S. Kim, and S. Lee, "Fast pu skip and split termination algorithm for hevc intra prediction," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 25, no. 8, pp. 1335–1346, Aug 2015.
[22] A. Jimenez-Moreno, E. Martinez-Enriquez, and F. Diaz-de Maria, "Mode Decision-Based Algorithm for Complexity Control in H.264/AVC," *Multimedia, IEEE Transactions on*, vol. 15, no. 5, pp. 1094–1109, Aug 2013.
[23] E. Martinez-Enriquez, A. Jimenez-Moreno, and F. Diaz-de Maria, "An adaptive algorithm for fast inter mode decision in the H.264/AVC video coding standard," *Consumer Electronics, IEEE Transactions on*, vol. 56, no. 2, pp. 826–834, May 2010.
[24] "HEVC reference software HM13.0." [Online]. Available: http://hevc.hhi.fraunhofer.de/
[25] F. Bossen, "JCTVC-F900 Common test conditions and software reference configurations," Fifth meeting of the Joint Collaborative Team on Video Coding (JCT-VC) Torino, IT, 2011.
[26] G.Bjontegaard, "Calculation of Average PSNR Differences Between RD-Curves," ITU-T, VCEG-M33, Apr. 2001.